Software Granularity
Intrinsic and extrinsic identifiers to the rescue

Roberto Di Cosmo

April 21st, 2021
Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- **30 years** of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- **20 years** of Free and Open Source Software
- **10 years** building and directing structures for the common good

1999 *DemoLinux* – first live GNU/Linux distro

2007 *Free Software Thematic Group*

- 150 members
- 40 projects
- 200Me

2008 *Mancoosi project* [www.mancoosi.org](http://www.mancoosi.org)

2010 *IRILL* [www.irill.org](http://www.irill.org)

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

In RDA: *Software Source code IG* (co-chair), *Source Code Identification WG* (co-chair)
Software Source code: pillar of Open Science

A plurality of needs

**Researcher**
- archive and reference software used in articles
- find useful software
- get credit for developed software
- verify/reproduce/improve results

**Laboratory/team**
- track software contributions
- produce reports / web page

**Research Organization**
- know its software assets
  - technology transfer
  - impact metrics
Software is not (just) data

Software has multiple facets in research

- a **tool**
- a **research outcome** or result
- the object of **study**

Source code is **special**

Software **evolves** over time

- projects may last decades
- the **development history** is key to its **understanding**

Layers of **complexity**

- **millions** of lines of code
- large **web of dependencies**
- sophisticated **developer communities**

Matplotlib library

Python dependencies

Real dependencies

Fake OS dependencies induced by package granularity

Roberto Di Cosmo @rdicosmo
What is software?

**Software as a concept**
- software project / entity
- the creators and the community around the project
- the software solution / functionality

**Software artifact**
- the binaries for different environments
- the software source code for each version
  - the multiple files or code fragments

**Versioning, granularity**

- **Project** “Inria created OCaml and Scikit-learn”
- **Release** “2D Voronoi Diagrams were introduced in CGAL 3.1.0”
- **Precise state of a project** “This result was produced using commit 0064fbd…”
- **Code fragment** “The core algorithm is in lines 101 to 143 of the file parmap.ml contained in the precise state of the project corresponding to commit 0064fbd….”
Granularity and identifiers

Extrinsic identifiers are key for the concept layers
Intrinsic identifiers are key for the artifact layers
In some cases, extrinsic identifiers can be added too

Top concept layers vs. bottom artifact layers
Granularity and identifiers

Top concept layers vs. bottom artifact layers

Extrinsic identifiers are key for the concept layers
Intrinsic identifiers are key for the artifact layers
In some cases, extrinsic identifiers can be added too
Granularity and identifiers

Top concept layers vs. bottom artifact layers

- Extrinsic identifiers are key for the concept layers
- Intrinsic identifiers are key for the artifact layers
- In some cases, extrinsic identifiers can be added too
Granularity and identifiers

Top concept layers vs. bottom artifact layers

Extrinsic identifiers are key for the concept layers
Intrinsic identifiers are key for the artifact layers
In some cases, extrinsic identifiers can be added too
Extrinsic and Intrinsic identifiers in a nutshell

Extrinsic identifiers: no per se relation with the designated Object

A register keeps the correspondence between the identifier and the object

pre-internet era passport number, social security number, ISBN, ISSN, etc.
internet era DOI, Handle, Ark, PURLs, RRID, etc.

Intrinsic identifiers: derived from the designated Object

No register needed to keep the correspondence between the identifier and the object

pre-internet era musical notation, chemical notation (*NaCl* is table salt)
internet era cryptographic hashes for distributed software development, Bitcoin

more in this dedicated blog post (with pointers to literature)
Meet the Software Heritage Identifiers (SWHIDs) (full spec)

An emerging standard
- in Linux Foundation’s SPDX 2.2
- IANA-registered "swh:" URI prefix
- WikiData property P6138

Examples
- Apollo 11 AGC excerpt
- Quake III rsqrt

for 20+ billions software artifacts!
Questions?

EOSC SIRS Task Force
Scholarly Infrastructures for Research Software
2020, European Commission, https://doi.org/10.2777/28598

Roberto Di Cosmo
Archiving and Referencing Source Code with Software Heritage
International Congress on Mathematical Software (ICMS), 2020

P. Alliez, R. Di Cosmo, B. Guedj, A. Girault, M. Hacid, A. Legrand, N. Rougier
Attributing and Referencing (Research) Software: Best Practices and Outlook From Inria

Roberto Di Cosmo, Morane Gruenpeter, Stefano Zacchioli
Referencing Source Code Artifacts: a Separate Concern in Software Citation
Computing in Science & Engineering, 2020, ISSN: 1521-9615

Jean-François Abramatic, Roberto Di Cosmo, Stefano Zacchioli
Building the Universal Archive of Source Code
Communication of the ACM, October 2018